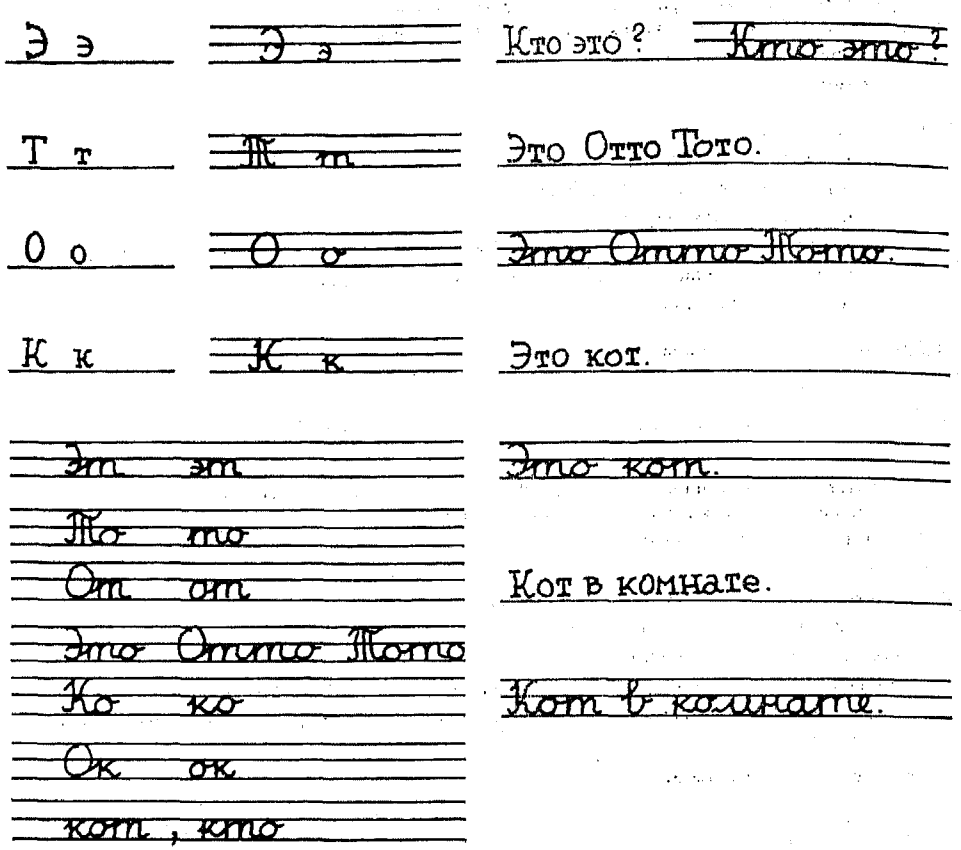
A tábla képe



Összegzés:

A célrendszer eredményes megvalósítását jól segítette a tankönyv olvasási szakaszának tanulást motiváló felépítése az írott betűk és szövegek megjelentetésével. A gyermek- és beszédközpontúság az írás tanításának szakaszában is érvényre jut, hiszen az új ismeretszerzés folyamatának a gyermekek aktív részesei, a folyamat minden mozzanata előzetes ismereteikre épül, a pedagógus tervező, szervező, irányító szerepe a mozzanatok sorrendjének meghatározásában érvényesül.

MÁGORINÉ HUHN ÁGNES
Szeged

Néhány általános iskolai feladat számítógépes megoldásáról

Napjainkban a számítástechnika korában érdekes és hasznos feladat lehet az általános iskolai feladatok számítástechnikai szempontból való elemzése, illetve feldolgozása. Sok feladat számítógéppel való megoldásánál ugyanazt az utat követ-

hetjük, mint a hagyományos feldolgozásnál, más feladatoknál viszont egészen eltérő algoritmusokat dolgozhatunk ki. Ennek bemutatására nézzünk néhány feladatot!

1. feladat: Határozzuk meg a 100-nál nem nagyobb 4-gyel osztható természetes számok összegét! (7. oszt. tankönyv 133. oldal 15. feladat.)

$$\text{Az } S_n = \frac{a_1 + a_n}{2} n = \frac{2a_1 + (n-1)d}{2} n$$

képletek valamelyikének alkalmazásával is megoldhatjuk a feladatot, amennyiben meghatározzuk a_n-et, vagyis a 4-gyel osztható, de 100-nál nem nagyobb természetes számok legnagyobbikát, vagy n-et, az 1 és 100 közé eső 4-gyel osztható természetes számok számát. Így feladatunk egy aritmetikai képletbe való helyettesítésre egyszerűsödik.

Ha a feladatot viszont úgy módosítjuk, hogy a 4-gyel osztható számok sorozatát is fel akarjuk írni, és a feladatot a fenti képletek alkalmazása nélkül akarjuk megoldani, (például 5. vagy 6. osztályban) a következő egyszerű algoritmust követhetjük. A sorozat tagjait rekurzív definícióval képezzük, s közben a tagok összegét is számítjuk.

Jelöljük az éppen számítandó tagot A-val, az addig kiszámított tagok összegét S-sel!

Nézzünk egy lehetséges algoritmust a feladat megoldására!

1. Az első 4-gyel osztható természetes szám a 0 és az eddig felírt 4-gyel osztható számok összege szintén 0.
   Így kezdetben A-nak és S-nek is 0 értéket adunk. (A=0 S=0).
2. Minden további A értéket az előzőből képzünk úgy, hogy 4-et hozzáadunk. (A= A+4).
3. Az addig képzett tagok összegéhez adjuk hozzá az új tagot! (S=S+A).
4. Vizsgáljuk meg, kell-e még a sorozatban további tagot képezni! Nézzük meg, hogy a következő tag még kisebb vagy egyenlő, mint 100! Ha igen, akkor a 2. ponttól ismételjük az algoritmus lépéseit, egészen addig, míg először azt nem találjuk, hogy a sorozat következő tagja már nagyobb lenne mint 100. Akkor ezt a tagot már nem kell képeznünk, tehát az eljárást befejezzük.

A következőkben a fenti algoritmust folyamatábrán szemléltetjük. Mivel a folyamatábra alapján számítógépes programot is akarunk írni, ezért a folyamatábrán feltüntetjük, hogy mikor íratjuk ki a számítógéppel a sorozat tagjait és a tagok összegét. A sorozat minden tagját a képzés után azonnal írassuk ki, míg az S értékét csak akkor, amikor már a képzendő összes tag összegét kiszámítottuk!

Folyamatábránk így a következő. (1. ábra).

Az eredmények kiíratása történhet papírra vagy display-re (tv-képernyőre), lehetőségeinktől függően.

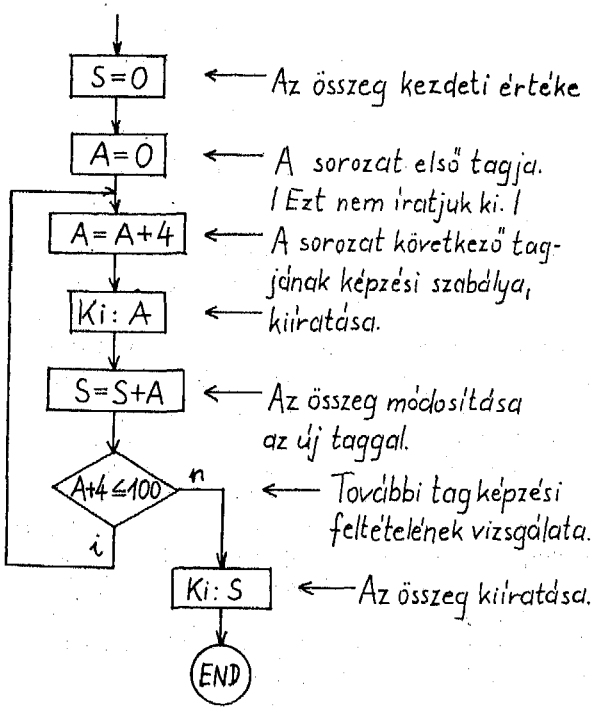
A feladat megoldása WANG számítógép BASIC programozási nyelvén:

```
10 S=0
20 A=0
30 A=A+4
40 PRINT A
50 S=S+A
60 IF A+4<= 100 THEN 30
70 PRINT "A SOROZAT TAGJAINAK ÖSSZEGE"; S
80 END
```

Vagy BASIC ciklus utasítást használva:

```
10 S=0
20 FOR A=4 TO 100 STEP 4
30 PRINT A
40 S=S+A
50 NEXT A
60 PRINT "A SOROZAT TAGJAINAK ÖSSZEGE"; S
70 END
```

A feladatot általánosíthatjuk: képezzük az N-nél nem nagyobb M-mel osztható természetes számok sorozatát és összegét! Az előbbiekben csupán annyit kell változtatni, hogy 100 helyett N-et írunk, 4 helyett pedig M-et. Az N és M így a feladat paraméterei, melyeket a számítógépnek a program futtatása során mondhatunk meg. (INPUT N, M a megfelelő BASIC-utasítás.)



1. ábra

2. feladat. 5. osztályban találkozunk a következő problémával: tízes számrendszerben adott természetes számot írjunk át például hármas számrendszerbe! (5. oszt. tankönyv 68. oldal.)

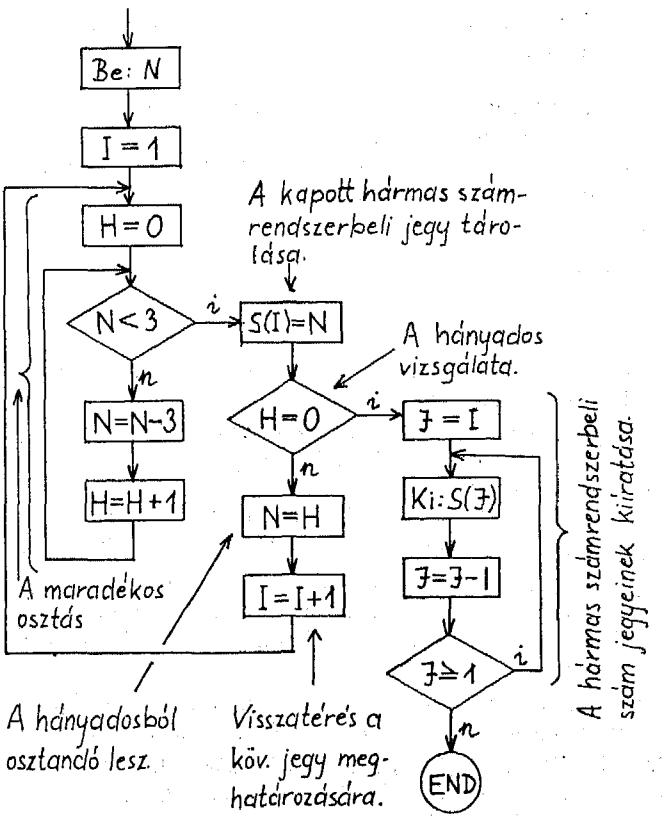
Az ismert algoritmusra tanítjuk meg a számítógépünket is. Az átírandó számot (N) elosztjuk 3-mal maradékosan. Bár legtöbb gépre erre kényelmes lehetőség van, mi minden gépre alkalmazható módszert követünk.

1. N-ből annyiszor vonjuk ki a 3-at, ahányszor lehet, (a természetes számok körében maradva) s közben számoljuk, hogy hányszor végeztük el a kivonást. (H)

Ez lesz a hányados, s az az érték, melyből a 3-at már nem lehetett levonni, a maradék. Ez a hármas számrendszerbeli szám első jegye. S(1). [Az S(I) vektorba tesszük az átírt szám számjegyeit.]

2. Ezután a hányadost osztjuk 3-mal maradékosan, feltéve, hogy a hányados nem 0. Így kapjuk a következő jegyet.
3. Az eljárást addig folytatjuk, míg a hányados 0 nem lesz.
4. S(I) tömb elemeit kell kiíratni, a nagyobb indexű elemektől haladva sorban S(1)-ig.

A fenti jelölésekkel folyamatábránk a következő lehet. (2. ábra).



2. ábra

Folyamatábránk alapján az alábbi BASIC-programot írhatjuk. Az átírandó számot INPUT N utasítással vihetjük be a gépbe.

```
10 DIM  S(100)
20 INPUT N
30 I=1
40 H=0
50 IF N<3 THEN 90
60 N=N-3
70 H=H+1
80 GOTO 50
```

```
 90 S(I)=N
100 IF H=0 THEN 140
110 N=H
120 I=I+1
130 GOTO 40
140 FOR J=I TO 1 STEP -1
150 PRINT S(J);
160 NEXT J
170 END
```

Megjegyzések:

1. A 10-es sorszámú utasítás az S(I) vektor számára biztosít helyet a memóriában.
2. 150 PRINT S(J); utasítás hatására a gép a számjegyeket egymás mellé írja. (A; miatt.)
3. A feladatot természetesen általánosabban is felírhattuk volna, ha 3 helyett tetszőleges $1<P\leqslant 9$ alapszámú számrendszerben gondolkodunk. A változtatás csupán annyi, hogy az 50-es és 60-as sorszámú utasításokban 3 helyett P-t írunk, s P-t is az N-nel együtt a 20-as sorszámú INPUT-utasítással olvassuk be a gépbe.

Tehát a változtatandó utasítássorok:

```
20 INPUT N, P
50 IF N<P THEN 90
60 N=N-P
```

A többi utasítássor változatlan.

3. feladat. Határozzuk meg az 1 és 100 közötti prímszámokat Eratoszthenészi szita segítségével! (6. oszt. tankönyv 57., 58. oldal.)

A számokat 1-től 100-ig írjuk egy A(I) vektorba.

(A(1)=1, A(2)=2, ... A(I)=I, ... A(100)=100)

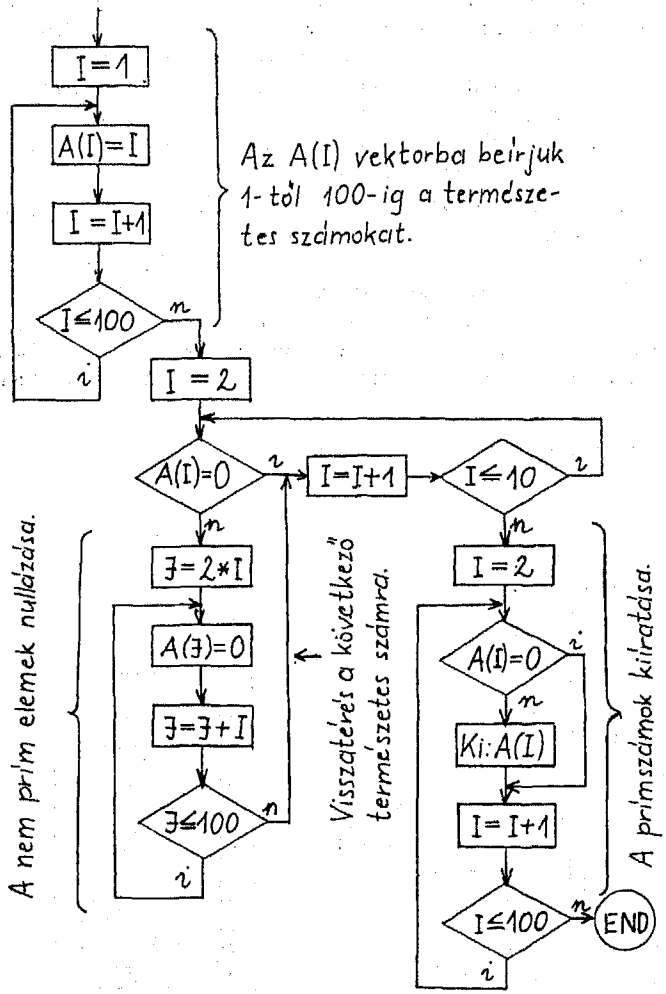
Az algoritmus a következő:

1. A második elemtől indulunk, s áthúzunk minden másodikat. A számítógépes megoldásnál az áthúzandó elem helyett írjunk 0-t!
2. Visszatérve a következő elemhez, megnézzük, át van-e húzva, 0-e? Ha nem 0, pl. P, innen indulva minden P-edik szám helyébe 0-t írunk. Ha 0 volt, megkeressük a következő nem 0 természetes számot, s 0-t írunk minden ennyiedik természetes szám helyett.
3. Ezt ismételjük, míg a vektor 10. eleméhez nem érünk. Továbbmenve újabb elemeket már nem nullázunk, mert bármely 100-nál nem nagyobb nem prímszámnak van $\sqrt{100}=10$-nél kisebb prímosztója, tehát már nulláztuk.
4. Végigvizsgálva a vektort a második elemétől a nem 0 elemek a prímek.

Írjuk fel a folyamatábrát! (3. ábra)

Ennek alapján a BASIC-program a következő.

```
 10 DIM A(100)
 20 FOR I=1 TO 100
 30 A(I)=I
 40 NEXT I
 50 FOR I=2 TO 10
 60 IF A(I)=0 THEN 100
 70 FOR J=2*I TO 100 STEP I
 80 A(J)=0
 90 NEXT J
100 NEXT I
```



3. ábra

```
110 FOR I=2 TO 100
120 IF A(I)=0 THEN 140
130 PRINT A(I);
140 NEXT I
150 END
```

Természetesen 100 helyett tetszőleges N-re is felírhattuk volna az algoritmust, figyelembe véve esetleges gépi megszorításokat.

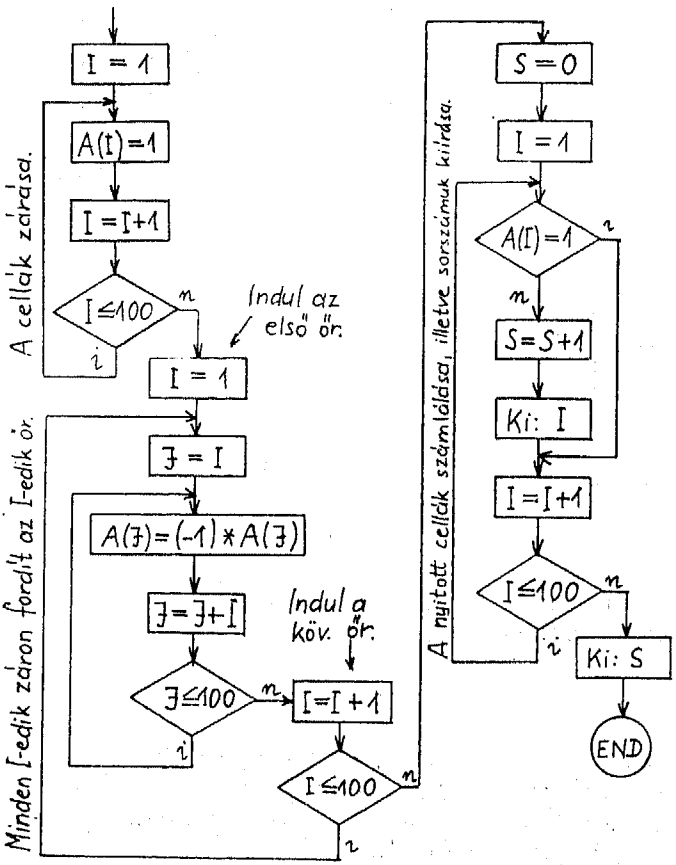
4. feladat. Nézzük a következő érdekes problémát, melyet a 6. osztályos tankönyv ismertet! (76. old. 53. feladat).

Egy szultán 100 cellába bezárat egy-egy rabot. A cellákon kétállású zárak vannak, forgatással felváltva nyílnak, illetve záródnak. A rabok nem veszik észre, ha

nyitják vagy zárják a cellákat. A száz rab bezárását követően a szultán meggondolja magát, és végigszalaszt egy őrt, hogy minden záron fordítson egyet, majd újra meggondolja magát, és elküld egy másik őrt, hogy minden második záron fordítson egyet, majd egy harmadik őrt, hogy minden harmadik záron fordítson egyet és így tovább. A századik őrt azzal a paranccsal küldi, hogy a századik záron fordítson egyet. Ezután elrendeli, hogy akinek a cellája nyitva van, azt bocsássák szabadon. Kérdés: mely cellákból és összesen hányan szabadulnak!

Az ismert számelméleti megoldás helyett szimuláljuk számítógéppel a fent leírt folyamatot! A celláknak egy 100 elemű vektort feleltessünk meg. A nyitott állapotban a vektor megfelelő eleme –1-gyel egyenlő, zárt állapotban pedig 1-gyel. A záron való fordítást így –1-gyel való szorzással érjük el. A –1-gyel való szorzást így először a vektor minden elemén végrehajtjuk, majd minden második elemet szorzunk –1-gyel, míg az I-edik lépésben csak minden I-ediket stb. Végezetül megnézzük, hogy a vektor mely elemei egyenlőek –1-gyel, és hány ilyen elem van. (Ezek száma S).

Írjuk fel az algoritmus alapján a folyamatábrát! (4. ábra).



4. ábra

Írjunk BASIC-programot is!

```
 10 DIM A(100)
 20 FOR I=1 TO 100
 30 A(I)=1
 40 NEXT I
 50 FOR I=1 TO 100
 60 FOR J=I TO 100 STEP I
 70 A(J)=(-1)*A(J)
 80 NEXT J
 90 NEXT I
100 S=0
110 FOR I=1 TO 100
120 IF A(I)=1 THEN 150
130 S=S+1
140 PRINT I
150 NEXT I
160 PRINT "A NYITOTT CELLÁK SZÁMA ="; S
170 END
```

Ha display-vel ellátott számítógép áll rendelkezésünkre, a feladat megoldását nagyon látványossá tehetjük, ha az algoritmus minden lépésének eredményét, vagyis minden őr munkáját kiíratjuk a display-re, és így a tanulók maguk előtt látják az egész folyamatot.

Azt hiszem, a bemutatott feladatok, melyek az általános iskolai tananyag igen különböző területeiről valók, már sejtetik azokat a lehetőségeket, melyek a számítástechnika általános iskolai alkalmazásában vannak.